Sources of Misinformation in Online Social Networks: Who to suspect?

Dung T. Nguyen, Nam P. Nguyen and My T. Thai

Department of Computer and Information Science and Engineering University of Florida

Email: {dtnghuyen, nanguyen, mythai}@cise.ufl.edu

Abstract—Online Social Networks (OSNs) have recently emerged as one of the most effective channels for information sharing and discovery due to their ability of allowing users to read and create new content simultaneously. While this advantage provides users more rooms to decide which content to follow, it also makes OSNs fertile grounds for the wide spread of misinformation which can lead to undesirable consequences. In order to guarantee the trustworthiness of content sharing in OSNs, it is thus essential to have a strategic investigation on the first and foremost concern: the sources of misinformation.

In this paper, we study k -Suspector problem which aims to identify the top k most suspected sources of misinformation. We propose two effective approaches namely ranking-based and optimization-based algorithms. We further extend our solutions to cope with the incompleteness of collected data as well as multiple attacks, which mostly occur in reality. Experimental results on real-world datasets show that our approaches achieve competitive detection ratios in a timely manner in comparison with available methods.

I. INTRODUCTION

The recently rising popularity of online social network sites, such as Facebook, Twitter and Google+, has created a huge impact on the landscape of information sharing and discovery nowadays. Due to the advantage of disseminating information to wider audiences, such OSNs enable their users to read and create new content at the same time, thus shifting the role of information broker from some dedicated news sources to a more diverse group of individuals. While this advantage allows online users more rooms to choose which content to follow, it also makes OSNs fertile grounds for the wide spread of misinformation which can lead to undesirable consequences. For example, the faulty swine flu tweets on Twitter caused a widespread panic in the general public in 2009 [1]. Similarly, the spread of misinformation about the health insurance reform gained a great concern in the public opinions [2], and even bothered the White House to publicly announce correction [3].

In order for online social networks to serve as a trustworthy channel for publishing important content, it is thus essential to have a strategic investigation on the core of misinformation spread. Perhaps the first and most crucial step in the investigation of such spreading is to trace back the sources of misinformation. Knowing these sources is very important to the network administrators in the following essences: (1) it helps to understand the ultimate goals for the misinformation, who the misleading information targets to as well as the potential size of the misinformation spread, and (2) it provides valuable insights into designing effective strategies for the

containment campaign. In fact, available strategies [4] [5] [6] for countering or containing misinformation are shown to be extremely effective under the knowledge of these sources. As a result, tracing back the originated sources is the heart of the counter misinformation game in OSNs. However, the huge magnitude of OSNs' users and their highly clustered structures makes this an extremely topical yet challenging problem.

A few attempts on this line of work have been proposed in the literature. In [7], Shah et al. study the problem of finding the sources of rumors on the SIR model using a maximum likelihood estimator. They extend their solution to general graphs via finding Breadth First Search (BFS) trees at different nodes. Since BFS tree is not unique, this methods might not be consistent especially when multiple sources exist. Lappas et al. [8] proposed the k -Effector problem which tries to find a set of at most k nodes that best explain the current activation state of the network. They suggest an optimal algorithm for this problem on trees, however, the extension of that algorithm does not appear to work well on general graphs. In [9], Qazvinian et al. propose a framework to detect people that endorse and further spread rumors out to a wider population. This method, however, requires a large amount of real training data for its proper function.

In this work, we put our focus more on the investigation process. In particular, we study the k -Suspector problem which aims to find the k most suspected users from the set of victims who are already influenced by the misinformation. The key differences between our work and prior studies [7] [8] are (1) we allow multiple sources spreading misinformation at the same time, and (2) our proposed solutions can work on a more general network with incomplete data and multiple attacks. In this line of work, we propose two efficient algorithms based on the *reverse diffusion process* and *ranking* of each user in order to hint out the suspected sources. Furthermore, we suggest a greedy algorithm with an approximation guarantee for k -Influence, a problem that is closely related to k -Suspector, and then use that solution to hint out the set of k most suspected sources. To validate our approach, we extensively conduct experiments on various types of networks with different structures. The empirical results show that our methods can trace the sources of misinformation with up to 80% accuracy, thereby implying that our approaches can enhance the reliability of information sharing in OSNs.

Organization: Section II describes the network model and k -Suspector problem. Section III presents our ranking-based

algorithms to identify the top k suspected sources of misinformation. In Section IV, we introduce another approach based on solving k -Influence problem. All proposed algorithms are evaluated on real networks in Section V. In Section VI, we study k -Suspector in the presence of multiple attacks and incompleteness of data. Section VII concludes our work.

II. PRELIMINARIES AND PROBLEM DEFINITION

Graph notations

Let $G = (V, E, p)$ be a directed graph representing a social network where V is the set of $|V| = n$ nodes (or users), E is the set of $|E| = m$ directed connections (or relationships, friendships), and $p = \{p_{uv}\}$ is the (normalized) weights associated with all edges (u, v) 's. In our model, p_{uv} is also interpreted as the probability that v gets influenced by u . Let N_u^- (N_u^+) be the set of incoming (outgoing) neighbors of node u . Finally, denote by A the set containing the original attackers, i.e., the sources of misinformation.

Diffusion model

We adopt the Independent Cascade (IC) model, a well-known model that has been widely accepted in the literature [10]. In this model, we call a node *active* (or infected) if it is exposed to the misinformation and *inactive* otherwise. The misinformation propagation is initiated from the source set A and continues over time. Any node u that becomes active in time t has only one chance to activate each of its currently inactive neighbor v in time $t+1$ with the successful probability p_{uv} . If v has multiple active neighbors, they will try to activate v once sequentially in an arbitrary manner. If any of these activations succeeds, v is activated and the same procedure continues on v . The propagation stops when no new nodes are activated.

Problem definition

Definition 1 (k -Suspector): Given a directed weighted graph $G = (V, E, p)$ representing a social network together with the IC diffusion model and a set of infected nodes I , our goal is to find a set S of k suspected nodes such that the number of original attackers in S is maximized.

III. RANKING-BASED APPROACHES

In this section, we discuss two algorithms for the k -Suspector problem based on the ranking of suspect level of a node. In a big picture, the first algorithm starts out from I and reversely traces back the suspected nodes in a reverse manner. The second approach iteratively ranks the suspect level of every active node by using the Monte Carlo method and returns the top k most suspected ones.

A. An Imeter-based algorithm

Misinformation which is initiated from the set of attackers A can only spread out to a larger population by exploiting the connections from the active nodes to their neighbors. Therefore, if we can derive the reverse process, we can definitely trace back those sources of misinformation from

the active set. This intuition motivates our first algorithm and is depicted in Fig. 1(a). In this figure, u_3 is the source of misinformation that initially activates u_5 , and u_5 next activates u_6 and u_8 . From the set of victims $I = \{u_3, u_5, u_6, u_8\}$, we can guess the possible attackers as follows: Node u_6 and u_8 share the only infected neighbor u_5 in common, thus they either are attackers or get activated by u_5 . Next, u_5 can in turn be either the victim of u_3 or an attacker itself. Now, u_3 has to be an attacker since there is no active node that can possibly influence it. In fact, what we have guessed is a simplification of the *reverse processes* that start out from u_6 and u_8 and then merge at u_5 . This merged process in turn traces back to u_3 and finally identifies u_3 as an attacker since u_3 has no direct link from active nodes. The real situation, of course, is much more complicated and challenging to track back the attackers since each active node can either be an attacker or just simply influenced by one of its neighbors (Fig. 1(b)). We next formally define the Reverse Diffusion Process and use it to identify the k suspected nodes.

Fig. 1. Two examples of misinformation propagation. In (a), it is easy to guess u_3 is the attacker. In (b), it is much more complicated to guess which nodes are the attackers.

Reverse Diffusion Process (RDP). Given a social network $G = (V, E, p)$ and the set of active nodes $I \subseteq V$, the reverse diffusion process starts out with $|I|$ reverse flows from active nodes in I . In each time step, a reverse flow either stops at the current node u or advances to one of u 's active incoming neighbors. Since an active neighbor $v \in N_u^- \cap I$ of u can activate u with probability p_{vu} , the probability that there are no neighbors activating u , or in other words, the probability that the flow stops at u , is :

$$P(\text{stop}(u)) = \prod_{v \in N_u^- \cap I} (1 - p_{vu})$$

If this reverse flow does not stop at u , it can further randomly advance to one of u 's active neighbors with probability proportional to the influence of that node on u . Thus, the probability that the flow advances from u to v is:

$$P(u \rightarrow v) = \frac{p_{vu}}{\sum_{z \in N_u^- \cap I} p_{zu}}$$

Additionally, when two or more reverse flows join together, they are combined into a new reverse flow. The whole reverse diffusion process terminates when there are no active reserve flows. A key observation from the RDP is that *the more times a node is hit by a reverse flow, the more likely it is an attacker*. Therefore, we introduce a new metric Imeter, which

counts the expected number of hits, to evaluate the suspect level of a node, and the higher Imeter of a node indicates the higher chance it is an attacker. In our first algorithm, Imeter is estimated by averaging the RDP on the network after R runs, where R is a predefined parameter. Based on the new metric Imeter, we propose Imeter-Sort, algorithm that selects k nodes with the highest Imeter's as the suspected set.

Algorithm 1 Imeter-Sort

Input: The social network $G = (V, E, p)$, the set of infected nodes I and k
Output: Top k suspected nodes.
 $R \leftarrow \#simulations$
Do R simulations of the Reverse Diffusion Process to compute $Imeter$ of all nodes.
Sort nodes in order $Imeter(u_1) \geq Imeter(u_2) \geq \dots \geq Imeter(u_{|I|})$
return k nodes $u_1, u_2, \dots, u_k$

B. Influence-based algorithm

In this section, we present another metric that can be used to measure the suspect level of a node. The intuition behind this algorithm is as follow: *for any node u , if u were the only source of misinformation then the more nodes in I that u can activate, the more likely it is an attacker*. Follow up with this intuition, we define $\mathcal{I}(u, I)$ as the influence of a node u on the set of active nodes I , i.e., the expected number of nodes in I that can be activated by only u . Our strategy is to identify k nodes u 's having the highest $\mathcal{I}(u, I)$'s as the k most suspected nodes.

To estimate $\mathcal{I}(u, I)$ with high accuracy, we utilize the Monte Carlo method in which R independent cascade processes are simulated to compute the average number of active nodes in I when $\{u\}$ is the initial active set. After estimating the influence of all nodes on I , we choose top k nodes to the suspected set as described in Algorithm 2. In terms of time complexity, it takes $O(m)$ to simulate an independent cascade process and $|I| \log |I|$ to sort nodes, so the total running time is $O(|I|mR)$. Note that this time complexity is linear in term of the number of connections in the network as well as the number of active nodes.

Algorithm 2 Influence-Sort

Input: The social network $G = (V, E, p)$, the set of infected nodes I and k
Output: Top k suspected nodes.
 $R \leftarrow \#simulations$
for Each u in I **do**
 Do R simulations of independent cascade processes to compute $\mathcal{I}(u, I)$
end for
Sort nodes in order $\mathcal{I}(u_1, I) \geq \mathcal{I}(u_2, I) \geq \dots \geq \mathcal{I}(u_{|I|}, I)$
return k nodes $u_1, u_2, \dots, u_k$

IV. OPTIMIZATION-BASED APPROACH

In this section, we provide another point of view into the k -Suspector problem. As one of the ultimate goals of misinformation spread is to attract as many users to as possible, the originated attackers should have the ability to either influence a lot of people or influence users who can in turn spread misinformation out to many more people. In both cases, the knowledge of those high influential users in I provides us valuable insights into where the sources of misinformation can

be. Exploiting this observation, we consider the set of k nodes that have highest influence on the active set I as the suspected subset. This subset can be obtained by solving the following optimization problem.

Definition 2 (k -Influence): Given a weighted directed graph $G = (V, E, p)$ representing a social network, the IC diffusion model and a set of infected nodes I , the goal is to find a set of k suspected nodes $S \subseteq I$ so that the expected number of active nodes in I activated by S , $prog(S) = E[|IC(S) \cap I|]$, is maximized. Here $IC(S)$ is the random set of nodes activated by S on the whole network.

Since the Influence Maximization problem [10] is a special case of the k -Influence problem when $I = V$, the NP-hardness of the former problem immediately implies the NP-hardness of k -Influence (Lemma 1). As a result, it seems unrealistic to look for an optimal solution for k -Influence since it may take exponential time to find. Therefore, we propose an $(1 - 1/e - \epsilon)$ -approximation algorithm that provides a set of suspected nodes while guaranteeing the quality of the solution is close to the optimal one.

Lemma 1: The k -Influence problem on Independent Cascade model is NP-hard.

Based on the framework of sumodular function, we design an $(1 - 1/e - \epsilon)$ -approximation algorithm for the k -Influence problem. Nemhauser et al. [11] proved that if the objective function is submodular and monotone then the greedy hill-climbing algorithm approximates the optimum solution within a factor of $(1 - 1/e)$. A function f is said to be submodular if it satisfies the “diminishing returns”, i.e., for all element u and pair of sets $C \subset T$, $f(C \cup \{u\}) - f(C) \geq f(T \cup \{u\}) - f(T)$. By decomposing $prog(C) = \sum_{IG \in R(G)} Prob(IG) prog_{IG}(C)$, where IG and $prog_{IG}(C)$ are the random influent graph and the probability that IG is generated via the precede random process, respectively, we prove that $prog()$ function is submodular and monotone increasing (Lemma 2). Since the estimation of $prog(C)$ function is time consuming [10], we use the Monte Carlo method to estimate it with arbitrary small error ϵ . This results in the approximation factor $(1 - 1/e - \epsilon)$ for the k -Influence problem.

Lemma 2: $prog_{IG}()$ is submodular on the graph IG . Consequently, $prog(C) = \sum_{IG \in R(G)} Prob(IG) prog_{IG}(C)$ is also submodular. In addition, $prog()$ is monotone nondecreasing on IC model.

Next, we present Influence-Greedy, the approximation algorithm for the k -Influence problem, to find a set of k suspected nodes in Algorithm 3. In this algorithm, $IC(S)$ denotes the random set of activated nodes by S according to the IC model. To compute the increment of the $prog()$ function when adding node u to S , we use the Monte Carlo method to simulate the process R times to get the average value of $IC(S)$. Since each simulation consumes $O(m)$ time, the total running time is $O(mRk|I|)$. This time complexity is linear in term of the number of edges and the number of activated nodes.

Using Lemma 2, we can show the following Theorem

Theorem 1: For the k -Influence problem, the Influence-

Algorithm 3 Influence-Greedy

Input: Influence graph $G = (V, E, p)$, set of infected nodes I .

Output: Top k suspected nodes.

```
 $S \leftarrow \emptyset$   
 $R \leftarrow \#simulations$   
for  $i = 1$  to  $k$  do  
  for each node  $u \in I \setminus S$  do  
     $S_u \leftarrow 0$   
    for  $j = 1$  to  $R$  do  
       $S_u \leftarrow S_u + |IC(S \cup \{u\}) \cap I|$   
    end for  
     $S_u \leftarrow S_u / R$   
  end for  
   $S \leftarrow S \cup \arg \max_{u \in I \setminus S} \{S_u\}$   
end for  
return  $S$ 
```

Greedy algorithm approximates the optimum solution within the factor of $(1 - 1/e - \epsilon)$.

V. EXPERIMENTAL RESULTS

In this section, we report the experimental results for the k -Suspector problem on four social networks including the Zachary's Karate Club (Zachary) [12], Epinions [13], Slashdot [14], and Facebook [15] whose descriptions can be found in the provided references. The statistics of these datasets are summarized in Table I.

A. Experiment setup

In our experiment, the activation probability associated with each edge (u, v) is $1/d_v^-$ where d_v^- is the in-degree of v . In each experiment, 10 nodes from the top 30% most influential nodes are randomly selected as attackers, and then the set of nodes I infected by attackers is obtained by simulating the independent cascade process. This 10-attacker setup is chosen since (1) in practice, there are not usually too many sources spreading misinformation at the same time, and (2) to make the detection more challenging since the sizes of test networks are way larger than that of the sources. We repeat the process 1000 times and report the averaged detection rate, i.e., the percentage of originated attackers contained in the discovered set as well as the running time of all algorithms. Moreover, in all algorithms that use the Monte-Carlo method, the number of simulations is set to 10000.

The efforts that are most relevant to our work are [7] and [8] and thus, we plan to compare our results with theirs. However, the algorithm proposed in [7] works on only one source at a time, and on top of that, it is under the SIR model which is different from ours. Therefore, we evaluate our approaches in reference to the Distance-Sort algorithm [8], which computes the expected influence vector of each node and then sorts nodes based on the distance from the corresponding expected influence vector to the infected vector.

B. Comparison with optimal solutions

We first evaluate the performance of our approaches on the smallest social network Zachary where the optimal solutions can efficiently be found. We consider two algorithms for finding optimal solutions: (1) Influence-Opt: provides the optimal solution for k -Influence problem by brutal force search, (2)

TABLE I
STATISTICS OF DATASETS

Name	Type	Nodes	Edges
Zachary	Undirected	34	78
Facebook	Undirected	90,269	1,823,331
Epinions	Directed	75,879	508,837
Slashdot	Directed	77,360	905,468

Distance-Opt: provides the set of k nodes whose expected influence vector has minimum 1-norm distance to the infected vector [8]. In this case, the numbers of sources and suspected nodes are set equal and ranging from 1 to 6. The detection results are shown in Figure 2.

Fig. 2. Optimal solutions on Karate Club dataset

As shown in Figure 2, Influence-Opt Alg. achieves the best detection ratio while Influence-Sort and Imeter-Sort Alg. follow up immediately with competitive results. When the number of attackers is just 5, the set of suspected node provided by Influence-Opt Alg. even contains 60% of attackers. This promising ratio implies that the optimal solution of the k -Influence problem is also a potential one for the k -Suspector problem. On the other hand, algorithms which are based on the distance between the expected influence vector and the infected vector provides considerably lower accuracies: the detected ratios of Distance-Sort and Distance-Opt are just 38% and 32%, respectively. These results confirm the effectiveness of our approaches in comparison with ones in [8].

C. Performance on large online social networks

We next report the performance of our proposed approaches in real OSNs. Algorithms to find optimal solutions are opted out since they do not appear to finish in a timely manner on those networks. The performance and running time of all algorithms is shown in Figure 3. As depicted in this figure, the detection rates of all methods tend to increase linearly as the number of suspect nodes k increases. In particular, Influence-Greedy Alg. achieves the highest detection rates in all experiments with up to 82%, 64% and 68% accuracy on Epinions, Facebook and Slashdot, respectively. Imeter-Sort and Distance-sort Algs. follow up with up to 77% (58%), 50% (49%) and 58% (60%) accuracy, respectively; while Influence-Sort Alg. is lag behind with roughly 50% detection accuracy as the highest. We observe that these detection rates for Influence-Greedy and Imeter-Sort Algs are quite impressive given the

Fig. 3. The detection rate (left) and running time (right) of proposed algorithms on Epinions (top), Facebook (middle), and Slashdot (bottom).

sizes of these networks and that there are only 10 originated attackers randomly scattered all over the graphs.

In terms of execution time, despite its impressive performance, Influence-Greedy Alg. is also the most time consuming methods overall. In general, its execution time tend to increase exponentially as k increases while other algorithms take fair amounts of time (from 2s to 8s) to complete their tasks. Although Ranking-based algorithms achieve lower detecting rates, their execution times do not depend on the number of suspected nodes k , and thus, they perform substantially faster than Influence-Greedy Alg. as the number of suspected nodes k increases. Among the other three approaches, Imeter-Sort and Influence-Sort Algs. usually require much less time than while the detection rates are highly competitive to Distance-Sort.

In summary, we observe that the Influence-Greedy Alg. can trace back the sources with very high accuracy, however, it might be time consuming on large networks. In contrast, Imeter-Sort and Influence-Sort Algs. perform highly competitively in a timely manner even on large social networks.

VI. MULTIPLE ATTACKS AND LOSS OF INFORMATION

A. Multiple attacks

In practice, there are chances that the same group of originated attackers can multi-spread misleading information by disseminating multiple waves of misinformation with different contents. This reassembles the concept of *multiple attack* scenario. As multiple waves of misinformation will create different sets of active nodes, we can exploit these additional pieces of information to discover a better set of suspects by extending our proposed algorithms. For simplicity, we assume that there are two waves of attacks with two sets of active

TABLE II
PERFORMANCE OF ALGORITHMS ON SINGLE AND MULTIPLE ATTACKS

		Epinions	Facebook	Slashdot
Influence-Greedy	Single	81.5%	64%	67.7%
	Multiple	79.6%	81.9%	80.1%
Influence-Sort	Single	50.2%	37.5%	43.5%
	Multiple	76%	78.9%	80.7%
Distance-Sort	Single	58.2%	49.4%	60%
	Multiple	67.3%	48.7%	60.1%
Imeter-Sort	Single	76.9%	49.7%	57.6%
	Multiple	76.1%	49.9%	64.5%

nodes I_X and I_Y . Note that our extension can be applied naturally to the case of more than two attack waves.

Extended Influence-greedy algorithm. The key point of extended algorithm is that $prog(\cdot)$ function can be modified to include both active sets I_X and I_Y . Given a set S , the expected influence of S on I_X and I_Y is:

$$prog(S) = E[|\sigma(S) \cap I_X| + |\sigma(S) \cap I_Y|]$$

Extended Ranking-based algorithms. Given two active sets I_X and I_Y , we can compute rankings of nodes in regard to each set, and then sum up the ranking of each node to have the total ranking. In particular, the combined ranking of node u is $r_X(u) + r_Y(u)$ where $r_X(u)$ and $r_Y(u)$ are rankings of u with regard to I_X and I_Y , respectively.

B. Incompleteness of collected data

The underlying assumption that we rely on so far requires the complete knowledge of I , the set of nodes that already exposed to misinformation. However, in reality, we often do not know that data beforehand, or such knowledge can not be completely mined from the network structure. The incompleteness of collected data, as a result, makes the detection of sources much more difficult since the actual attackers might not be collected in I . To cope with this loss of information, we enhance the search space for the suspected nodes to the whole network instead of just the set of active nodes I . This enhancement, though simple, find itself very effective in providing a solution for the k -Suspector problem especially in the presence of missing data, as described in next subsection.

C. Evaluation of extended algorithms

1) *Multiple attacks:* To evaluate the efficiency of the extended algorithms in multiple attack scenarios, we compare the precision with the case that there is only one attack. The testbed is set up similarly to Section V with 10 nodes from the top 30% influential nodes as attackers and with 2 waves of misinformation spread.

Figure 4 reports the relative performance of our extended algorithms with the support of additional data from multiple waves of misinformation. Influence-Greedy Alg. is still the best one which, at $k = 20$, achieves the detection ratio of 79.6 %, 81.9% and 80.1% on Epinions, Facebook and Slashdot, respectively. Influence-Sort Alg. witnesses a huge improvement on Multiple Attacks and surpasses Imeter-Sort as well as Distance-Sort Algs. to become the second best method. Influence-Sort's accuracy is competitively close to Influence-Greedy Alg. in all three datasets with the detecting ratio of

76%, 78.9% and 80.7%. Imeter-Sort and Distance-Sort Algs. are lag behind the others as they do not appear to well-explore the additional pieces of information.

In multiple attack scenarios, Influence-Greedy and Influence-Sort Algs. observe a significantly improvement on their detection accuracy while Imeter-Sort and Distance-Sort Algs. only experience a slight change. This is reflected in Table II. Table II also compares the performance of all algorithms between single and multiple attacks with the set of 20 suspects. In general, the detection ratio of Influence-Greedy Alg. increases from 64% to 81.9% on Facebook and from 67.7% to 80.1% on Slashdot, respectively. On the hand, the results of Influence-Sort Alg. on multiple attacks outperforms those on single attack. The detection accuracy of Influence-Sort Alg. improves by 50% on Epinions and almost doubled on Facebook and Slashdot. In the single attack scenarios, this algorithm does not seem to capture sufficient information; however, with multiple attacks it is provided with much more information. Thus, its performance closely approximates that of Influence-Greedy Alg.

2) *Incomplete data*: We validate the accuracy of all algorithms when there is only a fraction of active nodes is known. In this experiment, each node in the active set I is randomly disregarded with probability 0.7, resulting in roughly 70% of data incompleteness. All algorithms, except Distance-Sort Alg., are performed on this incomplete set of active nodes to find most k suspected ones. The Distant-Sort Alg. is excluded from comparison since it requires the complete knowledge of I to compute the distance between the expected influential vector and the infected vector. The averaged detection ratios of all algorithms after 1000 runs are reported in Figure 4. These results show that Imeter-Sort is sensitive to the incompleteness of data: 30% of the remained information about the infected set is not enough for Imeter-Sort to track back to the sources of misinformation, which explains low detection ratio. Although Influence-Greedy and Influence-Sort Algs. are highly impacted by the incompleteness, they can still reveal up to 20% misinformation sources when the number of suspected nodes is 20. This implies that the Influence-based algorithms possess the ability to tolerate well with missing data.

VII. CONCLUSIONS

We study the k -Suspector problem aiming to narrow down the sources of misinformation in a social network. We propose two algorithms, namely ranking-based and optimization-based algorithms, for this problem. Our first algorithm starts out from the infected set and reversely traces back the suspected nodes. The second approach iteratively ranks the suspect level of every active node by using the Monte Carlo method and returns the top k most suspected ones. Furthermore, we study k -Influence, a problem that is closely relates to k -Suspector and suggest an approximation solution for it. Empirical results show that our approaches can traces back the sources of misinformation with high accuracy while a solution for k -Influence can also be regarded as an excellent hint for detecting those sources.

Fig. 4. Performance of extended algorithms on Epinions (top), Facebook (middle), and Slashdot (bottom).

REFERENCES

- [1] E. Morozov, "Swine flu: Twitters power to misinform," *Foreign Policy*, April 2009.
- [2] "First thoughts: Obama's good, bad news." http://firstread.msnbc.msn.com/_news/2009/08/19/4431138-first-thoughts-obamas-good-bad-news.
- [3] "Facts are stubborn things." <http://www.whitehouse.gov/blog/Facts-Are-Stubborn-Things/>.
- [4] C. Budak, D. Agrawal, and A. El Abbadi, "Limiting the spread of misinformation in social networks," in *WWW*, ACM, 2011.
- [5] N. P. Nguyen, G. Yan, M. T. Thai, and S. Eidenbenz, "Containment of viral spread in online social networks," in *WEBSCI*, 2012.
- [6] T. N. Dinh, D. T. Nguyen, and M. T. Thai, "Cheap, easy, and massively effective viral marketing in social networks: Truth or fiction?," in *Hypertext*, 2012.
- [7] D. Shah and T. Zaman, "Rumors in a network: Who's the culprit?," *Information Theory, IEEE Trans. on*, 2011.
- [8] T. Lappas, E. Terzi, D. Gunopulos, and H. Mannila, "Finding effectors in social networks," in *KDD*, ACM, 2010.
- [9] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei, "Rumor has it: Identifying misinformation in microblogs," in *EMNLP*, 2011.
- [10] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *KDD*, ACM, 2003.
- [11] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions - i," *Mathematical Programming*, vol. 14, pp. 265–294, 1978.
- [12] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of Anthropological Research*, vol. 33, no. 4, 1977.
- [13] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *The Semantic Web - ISWC 2003*, Lecture Notes in Computer Science, ACM, 2003.
- [14] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters," *Internet Mathematics*, 2009.
- [15] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Social Networks (WOSN'09)*, 2009.